ijecs open access www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 4 Issue 8 Aug 2015, Page No. 13828-13831

A Fuzzy Improved ACO Approach for Test Path Optimization

***Ritu*[1], *Ajay Dureja*[2]**
Student, PDM college of engineering and technology[1]
Bahadurgarh, Haryana
Assistent Professor, PDM college of engineering and technolog[2]
Bahadurgarh, Haryana

***Abstract*— To improve the software reliability it is required to test each aspect of software system. But sometime, the testing also increases the cost and time of software release. Because of this, it is required to execute the only the effective test cases undere some controlled mechanism. In this paper, a fuzzy improved ACO approach is defined to generate the optimized test path. To assign the weightage to test cases, the module interaction analysis and fault analysis are used collectively in this work. The experimentation is performed on some real time programs. The results shows that the proposed approach has reduced the overall test cost cost of software system**

Keywords: Path Testing, Black Box, ACO, Fuzzy, Module Interactivity

INTRODUCTION

Software Path Testing is able to select the required test cases from the pool based of different criteria so that the software test will be optimized without affecting the software quality. The software test path is identified under various aspects and component analysis along with requirement case identification[1][2][3][4]. The analysis is also defined under certain set of requirements. When the software system is defined at the earlier stage, the software system is divided in smaller software modules. These module are defined under various matrices and the interaction between the modules. The individual module analysis is required to prioritize the module and its requirement. The requirement phase also depends on the under different vectors associated to different stakeholders. It means, the software module having the importance for software tester or developer is not necessary to considered at same level by the end user. Because of this, different metrics are defined respective to different user types to assign the priorities to the software system[5][6][7][8]. Some of these metrics are shown in figure 1.

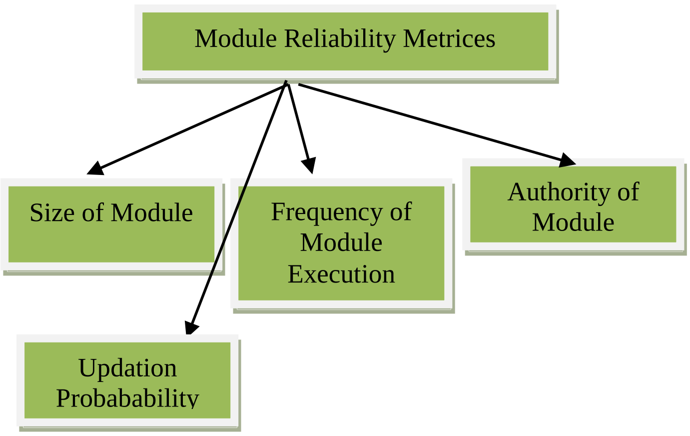

Figure 1 : Module Reliability Metrics

The foremost vector of the software system is the size metrics. The size can be defined in terms of collective or individual module or respective to number of options or LOC inclusive in the module. Larger the module size, higher the priority can be. Another user prespective to the module reliability is the frequency of use for end user. A module which is been used in daily routing requires higher reliability and efficiency. The problem in such module can be notice easily. The problem in such module cannot be bearable. Higher the module execution frequency, more critical the reliability of software module will be[9][10][11]. Another critieria associated with software module analysis is the authority of module itself. A module which is available at management level is considered to provide more effective, accurate and reliable results. The module provided to the third party can be comparatively lesser imporantace. Another vector that represents the changibility in the software system is the probability measure under updation vector. A software system which is updated regularly is considered more critical then static module[13].

To provide the software test plan, it is required to first analyze the software system under these aspects as well as identify the associated test cases with these software module. While defining the test cases, it is required to define the relative parameters and associated constraints so that the test path optimization will be done. The basic design of a software test plan is shown in figure 2.

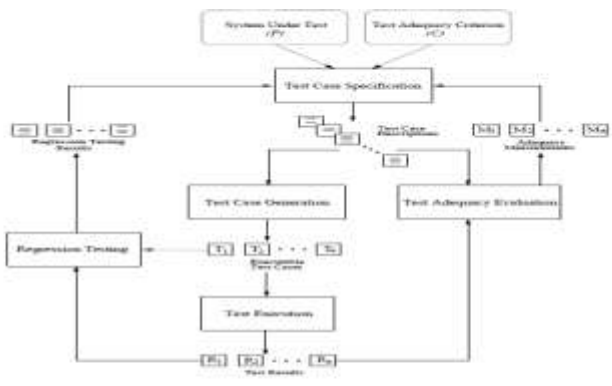

Figure 2 : Test Plan

Here figure 2 is showing the software test plan for the software system. This plant is based on the test path analysis under critiera specification , adequancy evalaution. Once the relative test cases are specified, the proper test plan is defiend alogn with description of individual test cases. These test cases are also described relative to the software modules and the test case specification. These test cases are then executed to provide the effective test results. The figure has shown the basic test plan applied by any black box testing approach.

In this paper, an optimized test path generation mechanism is presneted using ACO and fuzzy approach. The criteria selected as the parameter for test case generation includes fault analysis, module interaction analysis etc. In this section, an exploration of base test plan is defined. The section has prsented the clear view to analyze the software system and to provide the test case generation plan. In section II, the proposed research methodology is presented. In section III, the results obtained from the work are presented and discussed. In section IV, the conclusion of the work is presented.

I. RESEARCH METHODOLOGY

In this present work, a reliable and effective test path optimization is presented using fuzzy adaptive ACO approach. The system is able to deliver more reliable software system in real time environment. The analysis is here done under two main vectors called fault adaptive analysis and module interactive analysis. The model is able to provide the optimal solution so that the selection of the model will be done. The stages of this model are shown in figure 3.

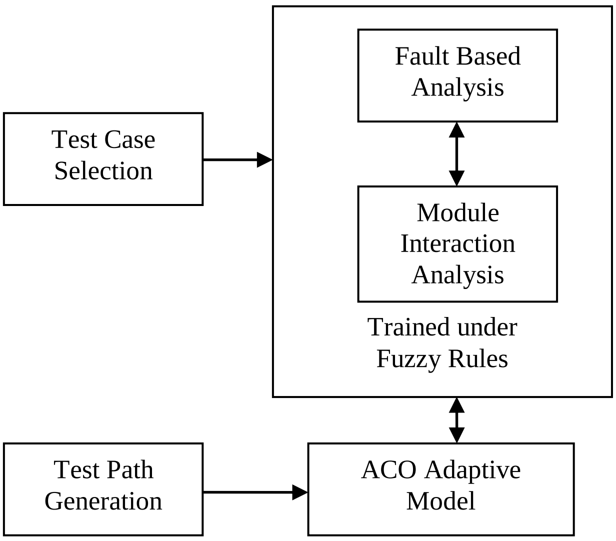

Figure 3 : Proposed Model

Here figure 3 is showing the proposed model for generating the optimized path for effective and reliable black box testing. As shown in the figure, the complete work is divided in two main stages called test case selection and test path generation. The test case selection is here done under two main vectors called fault adaptive analysis and module interaction analysis. These vectors are collectively trainined using fuzzy operators. Once the fuzzy is applied on these vectors, the priority of each test case and relative cost is assigned. After assigning the weights to different test cases over the software system it is taken by the ACO model as the population vector. This population set is then processed under ACO approach to generate the optimized test path. The ACO approach is here applied to analyze the priorty adaptive level cost. To generate this cost, the random ants are distribted over the network. The network is here considered as the relational matrix obtained from the test interactivity. These random

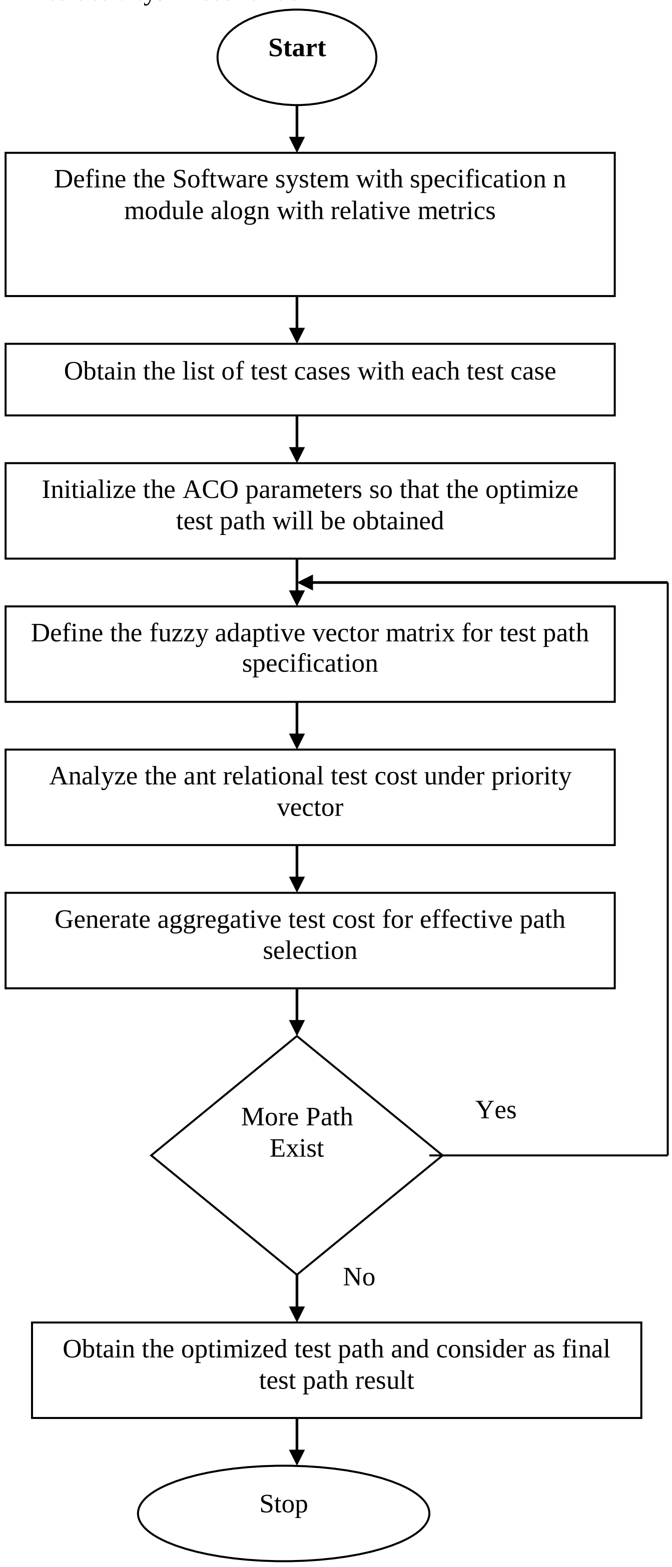

Figure 4 : Test Path Generation

DOI: 10.18535/ijecs/v4i8.38

ants analyze the relational analysis under defined constraints and obtain the effective relational path. The analysis is also performed to avoid the duplicate test cases so that the optimum path will be obtained. The least cost path is obtained from the work. The algorithmic model of this work is shown in figure 4.

The figure 4 is showing the flow of the test path generation process defined integratedly using ACO and fuzzy approach. The fuzzy is here defined as the vector or metric based on which the cost of a test case can be defined. Once the cost effective test paths are generated, the next work is to perform the test path adaptation under multiple vectors. This vector is defined for test path optimization. ACO approach is here defined to analyze the all the test cases partially ant restriction. Once the individual paths are obtained, the collective path is generated and the final optimized result is presented as the solution path.

IV. RESULTS

The presented work is implemented on real time software programs. The programs can be defined in any language but to take the example, a c++ driven program is

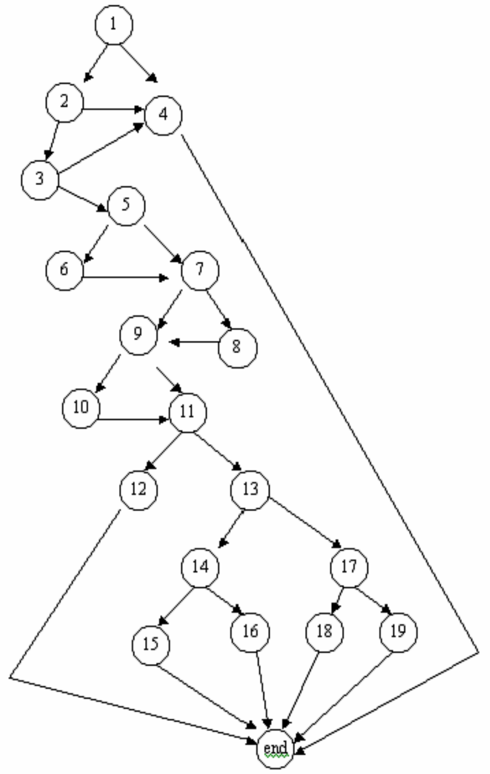

Figure 5 : Flow of Software System

considered in this work. Once the program is obtained, it is presented in the form of path flow. In this stage, the associated module based representation and the module interactively is defined. This module interactivty is also able to generate the optimized test path for the software system. The flow of the considerd software system is shown in figure 5.

In this figure, the system is defined with 20 different module alogn with module interaction. The fuzzy adaptive ACO model is applied for path optimization under fault analysis, module interaction analysis and collective form. The results are here taken in the form of path cost analysis. The obtained results are shown in the figure 6.

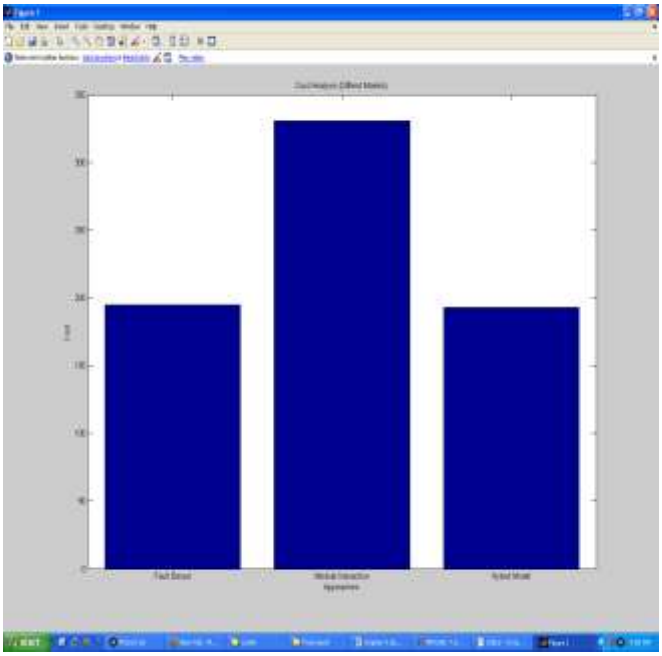

Figure 6 : Cost Adaptive Analysis

Here figure 6 is showing the cost based analysis of these three weighted vector applied under same algorithmic approach. The figure is showing that the proposed hybrid model has reduced the overall cost of software path generation.

V. CONCLUSION

In this paper an optimized module for black box testing is presented using fuzzy adaptive ACO approach. The model is based on collective analysis under two main criterias called fault analysis and module interaction analysis. The experimentation is applied on real time programs. The results shows that the work has generated the optimized test path for software system.

REFERENCES

[1] Williams L., "Testing Overview and Black-Box Testing Techniques", page no.35- 59, 2006.

[2] Pressman, Roger S.,"Software engineering: a practitioner's approach" 5th edition, 2001.

[3] Last M., Eyal S., and Kandel A.,"Effective Black-Box Testing with Genetic Algorithms",2005

[4] Singh K., Kumar R., "Optimization of Functional Testing using Genetic Algorithms" ,International Journal of Innovation, Management and Technology, Vol. 1, No. 1, April 2010.

[5] Engström E., "Black Box Test Selection and Product Line System Testing", Third International Conference on Software Testing,

Verification and Validation, 978-0-7695-3990-4/10 $26.00 © 2010 IEEE

[6] Jin W., Orso A., "Automated Behavioral Black Box Testing", Third International Conference on Software Testing, Verification and Validation 978-0-7695-3990-4/10 © 2010 IEEE

[7] Do H., Mirarab S., "The Effects of Time Constraints on Test Case Prioritization: A Series of Controlled Experiments", IEEE Transaction on Software engineering, vol. 36, no. 5, September, 0098-5589/10/ 2010 IEEE

[7] Foo K., Jiang Z., Adams B., "Mining Performance Black Box Testing Repositories for Automated Performance Analysis",10th International Conference on Quality Software, 1550-6002/10 © 2010 IEEE

[8] Zhang C., *et al.* "An Improved Black Box Test Selection Technique by Clustering Execution Profiles", 10th International Conference on Quality Software 1550-6002/10 © 2010 IEEE

[9] Kumar A., Tiwari S., Mishra K., "Generation of Efficient Test Data using Path Selection Strategy with Elitist GA in Black Box Testing", 978-1-4244-5540-9/10 ©2010 IEEE

[10] Li B., *et al.*, "Automatic Test Case Selection and Generation for Black Box Testing of Composite Service Based on Extensible BPEL Flow Graph", and 60773105, and partially by National High Technology Research and Development, 26th international conference on software maintenance, 978-1-4244-8628-1/10 ©2010 IEEE

[11] Kumar M., *et al.,"* Requirements based Test Case Prioritization using Genetic Algorithm", IJCST Vol. 1, Issue 2, December 2010.

[12] Rothermel G., Roland H.," Test Case Prioritization: An Empirical Study", International Conference on Software Maintenance, Oxford, UK, September, 1999, IEEE Copyright.

[13] Rus I., *et al.*, "Software Dependability Properties: A Survey of Definitions, Measures and Techniques". Fraunhofer Technical Report 03-110, January 2003.